Biotic Prediction

Building the Computational Technology Infrastructure
for Public Health and Environmental Forecasting

Software Engineering / Development Plan

BP-SEP-1.2

Task Agreement: GSFC-CT-1

September 26, 2002

Milestone A: Software Engineering Plan Completed, Due 04/15/02

The following documentation shall be provided in fulfillment of this milestone:

- Title of the agreement and agreement number
- Text of the milestone and its due date
- Contact information for the lead software engineer and team members
- A written description of the application being developed
- Preliminary requirements definition for the application
- Strategy for phased approach to software development for the full lifecycle of the model, linked to the requirements
- Design elements that enable ease-of-maintenance and robust integration of experimental modules
- Plans for open source, software reuse, portability, and interoperability with other community efforts
- Plans for managing the software configuration and controlling multiple versions of the software
- Plans for ensuring that accepted (and documented) software practices and processes are adopted and followed (quality assurance)
- Plans for community engagement beyond delivery and receipt of comments
- Plans to collaborate with the CT Evaluation Team's efforts to instrument development codes
- Evaluation/audit process
- Software maintenance plan
- Validation plan
- Risk assessment and mitigation plan

Contents

1 Overview **4**
1.1 Introduction . . . 4
1.2 Referenced Documents . . . 4
1.3 Document Overview . . . 4

2 Introduction **5**
2.1 Background . . . 5
2.2 Organization and Responsibilities . . . 6
2.2.1 Project Personnel . . . 6

3 Statement of the Problem **8**
3.1 Objectives . . . 8
3.2 Description of Application . . . 8
3.3 Preliminary Requirements . . . 11

4 Technical Approach **12**
4.1 Development Environment . . . 12
4.1.1 Security . . . 13
4.2 Community Engagement . . . 13
4.3 Reuse and Reusability Strategy . . . 14
4.3.1 Reuse . . . 14
4.3.2 Reusability . . . 14
4.4 Defect Tracking . . . 14
4.5 Software Lifecycle . . . 14
4.5.1 Requirements Analysis . . . 15
4.5.2 Preliminary Design . . . 16
4.5.3 Detailed Design . . . 16
4.5.4 Implementation . . . 17
4.5.5 Testing . . . 17
4.5.6 Maintenance . . . 17
4.6 Build strategy . . . 18
4.7 Tools and Technologies . . . 18

5 Management Approach **20**
5.1 Schedule . . . 20
5.2 Evaluation . . . 20
5.3 Risk Management . . . 20
5.4 Validation Plan . . . 20

6 Product Assurance **22**
6.1 Configuration Management . . . 22
6.2 Quality Assurance . . . 22

7 Plan Update History **23**
7.1 Version 1.2, September 26, 2002 . . . 23

A Contact Information **24**
A.1 Team Members 24
A.2 Document / System Access 25

B Glossary **26**

C Task Descriptions **27**

D Schedule Detail **30**

List of Tables

1 Referenced Documents 4
2 Detailed breakdown of tasks and anticipated responsibilities. 7
3 Task Descriptions 27

List of Figures

1 Context Diagram 5
2 Algorithm Diagram 9
3 Example map 10
4 Current Predictive Modeling Process 10
5 Hardware Development Environment 12
6 Software Lifecycle Waterfall 15
7 Schedule 20

1 Overview

1.1 Introduction

This project will develop the high-performance, computational technology infrastructure needed to analyze the past, present, and future geospatial distributions of living components of Earth environments. This involves moving a suite of key predictive, geostatistical biological models into a scalable, cost-effective cluster computing framework; collecting and integrating diverse Earth observational datasets for input into these models; and deploying this functionality as a Web-based service. The resulting infrastructure will be used in the ecological analysis and prediction of exotic species invasions. This new capability will be deployed at the USGS Midcontinent Ecological Science Center and extended to other scientific communities through the USGS National Biological Information Infrastructure program.

1.2 Referenced Documents

Table 1. Referenced Documents

Document Title	**Version**	**Date**
Configuration Management Plan	1.0	2002-04-08
Quality Assurance Plan	1.0	2002-04-08
Risk Management Plan	1.0	2002-04-08
Concept of Operations	1.0	2002-07-15
Software Requirements Document	1.0	2002-07-15
Software Design Document	TBD	TBD
Software Test Plan	TBD	TBD
Software Maintenance Manual	TBD	TBD
Software User's Guide	TBD	TBD

1.3 Document Overview

This document, the *Software Engineering / Development Plan,* describes our plan for developing the software for the *Invasive Species Forecasting System* (ISFS).

Section 2 provides an introduction to the project and a description of the project personnel and groups that will be involved with the project.

Section 3 has a description of the software application and a list of preliminary requirements for the software.

Sections 4 and 5 describe the technical and management approach to the development respectively.

Product assurance is discussed in section 6 and elaborated on in two other documents: the *Configuration Management Plan* and the *Quality Assurance Plan*.

This is intended to be a "living" document and will be updated throughout the project. At each major milestone, changes to this plan will be summarized in section 7.

Contact information for all team members is included in appendix A and a glossary of acronyms is in appendix B.

2 Introduction

2.1 Background

The primary product of this work will be a dynamic and flexible application system that will allow resource managers to integrate high-resolution satellite data with other types of ground-based data in order to model and analyze regional-scale biotic resources. We refer to the system as an "Invasive Species Forecasting System" (ISFS). As shown in Figure 1, the ISFS will ingest a wide range of data, including country and national-level plant and animal distribution data, point data, soil and atmospheric data, and remotely sensed data. The information products produced by the system will be predictive spatial statistical models and electronic and printed maps of potential "hot spots" of native plant diversity, including: (1) probable locations of rare habitats, (2) probable locations of relict species assemblages, (3) potential areas of future invasion, (4) spatial auto-correlations with cross-correlation statistics for single exotic species, (5) accuracy assessments of native and exotic plant diversity, (6) evaluation levels of uncertainty in maps of natural resources, and (7) classification and regression trees for map accuracy using multi-phase (double) sampling.

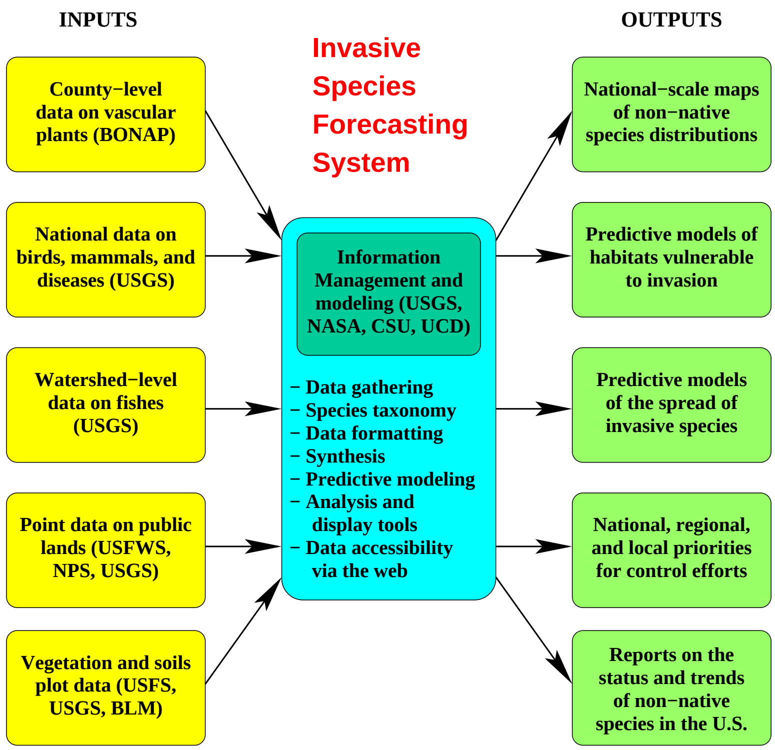


Figure 1. Illustration of how the Invasive Species Forecasting System fits into the overall USGS invasive species measurement and monitoring approach.

2.2 Organization and Responsibilities

2.2.1 Project Personnel

As shown in Table 2, this project will involve ecologists, computer scientists, engineers, and statisticians from NASA Goddard Space Flight Center (GSFC), the USGS Midcontinent Ecological Science Center (MESC), the USGS National Biological Information Infrastructure Program (NBII), and the Colorado State University Natural Resources Ecology Laboratory (NREL):

NASA GSFC:

- John L. Schnase (PI)
- James A. Smith (CI)
- John E. Dorband
- Jacqueline Le Moigne
- Jeffrey T. Morisette
- Jeffrey A. Pedelty
- Curt A. Tilmes

USGS MESC:

- Thomas J. Stohlgren (CI)

USGS NBII:

- Michael T. Frame (CI)

CSU NREL:

- Mohammed A. Kalkhan
- Robin M. Reich

Drs. Schnase (PI) and J. A. Smith (CI) will provide high-level guidance to the project and coordinate overall activities. Dr. Dorband will provide high-level guidance to the cluster computer infrastructure development and technology training activities. Dr. Pedelty will lead prototype code development and testing assisted by Drs. Dorband and LeMoigne. Mr. Tilmes will lead the system architecture design and integration component of the project and will coordinate the project's formal software engineering strategy and community deployment plan. Dr. Morisette will provide statistical expertise and assist in the development of geostatistical algorithms. In addition, we will contract with one or more software engineers who will be responsible for producing final code and documentation products.

Dr. Stohlgren (CI) will provide high-level guidance to the science component of the project and coordinate the activities of USGS and CSU. Dr. Kalkhan will provide training and modeling expertise and coordinate the acquisition, formatting, and delivery of data products for the project. Dr. Reich will provide statistical expertise and assist in the development of geostatistical algorithms. Mr. Frame (CI) will coordinate integration of products and services into the USGS NBII program and will facilitate community deployment activities.

Complete contact information for team members can be found in appendix A on page 24.

Table 2. Detailed breakdown of tasks and anticipated responsibilities.

		NASA GSFC								USGS/CSU				
	Lead	John L. Schnase	Jim A. Smith	John Dorband	Curt Tilmes	Jacqueline Le Moigne	Jeff Pedelty	Jeff Morisette	Software Engineer (TBD)	Tom Stohlgren	Mohammed Kalkhan	Data Manager (TBD)	Mike Frame	Robin Reich
Data Ingest														
Ancillary layers	MK									•	•	•		
Field data (points)	TS/MK									•	•	•		
Remote sensing (grids)	JM		•				•	•	•	•	•	•	•	
Automation (WWW)	TBD				•		•		•	•	•	•	•	
Preprocessing														
Quality Assurance	CT				•				•			•		
Formats (GIS,S+,Matlab)	CT				•				•			•		
Modeling components														
Model selection	MK/JM	•		•			•	•	•		•			•
Tests for autocorrelation	MK/JM	•		•			•	•	•		•			•
Modeling / kriging	MK/JP	•		•			•	•	•		•			•
Modeling / co-kriging	MK/JP	•		•			•	•	•		•			•
Web-based implementation	JLS/TS	•			•		•	•	•	•	•		•	
Mapping														
Summaries/description	TS	•					•			•	•			
Prediction	TS/JLS	•					•			•	•			
Web-based implementation	JLS/TS	•					•			•	•		•	
Reporting & Documentation														
Internet documentation	MF													
Briefing material	JLS													
Software Eng. Plan	CT	•	•		•		•			•				
Configuration Mgmt Plan	CT	•	•		•					•				
Quality Assurance Plan	CT	•	•		•					•				
Annual reports	JLS/JAS	•	•							•				
Final report	JLS/JAS	•	•							•				
New Users														
	TS													
Hardware														
GSFC	JD			•			•							
USGS	TBD													
Software														
Algorithms	RR			•		•	•				•			•
Prototyping	JP	•	•	•		•	•	•						•
Engineering	CT	•	•	•	•	•			•				•	

3 Statement of the Problem

3.1 Objectives

The specific objectives of this project include the following. We will:

- Develop and use a community standard for implementing high-performance, landscape-scale, predictive biological distribution models;
- Create a high-performance, parallel implementation of the USGS *PlantDiversity* (invasive species) model code;
- Document the use of software engineering techniques that foster reproducibility and community-wide software process improvements in these domains;
- Engage an extended community of scientists through the established NBII community infrastructure program; and
- Empower the ecological, environmental, and public health communities by expanding their participation in high-performance computing and greater use of NASA data.

3.2 Description of Application

Just testing changebars.

The Invasive Species Forecasting System is based on a spatial, geostatistical model called *PlantDiversity*. The *PlantDiversity* model, developed by colleagues at NREL and MESC, focuses on the ecological analysis and prediction of exotic species invasions. The model is recognized as both important and representative of how problems of this type are being approached by various biological, environmental, and ecological communities.

Predictive spatial models developed from multi-scale data are an excellent example of data synthesis for natural resource management and public health. Spatial statistics and geostatistics provide a means to develop spatial models that can be used to correlate coarse scale geographic information (e.g., digital elevation models, burn areas, remotely sensed data) with multi-scale field measurements of biotic and abiotic variables. Integral to the creation of spatial models is the collection of appropriate data. Drs. Stohlgren, Kalkhan, and Reich have developed a multi-phase, multi-scale sampling approach that involves stratification of areas of interest from remotely sensed data, random location of field sampling points within strata, and sampling with multi-scale plots. Data collection from multi-scale plots allows extrapolation of results to larger scales with calculable error.

The ability to model small-scale variability in landscape characteristics requires the generation of full-coverage maps depicting characteristics measured in the field. While many spatial datasets describing land characteristics have proven reliable for macro-scale ecological monitoring, these relatively coarse scale data fall short in providing the precision required by more refined ecosystem resource models. Spatial statistics and geostatistics provide a means to develop spatial models that can be used to correlate coarse scale geographical data with field measurements of biotic variables. This general landscape analysis approach is being used successfully to address a range of natural resource and public health issues.

Figure 2 summarizes the steps involved in the USGS modeling approach. The process begins with stepwise regression and trend surface analysis for geographical variables and measures of focal taxa to evaluate large-scale spatial variability in the study area. Once a spatial, or temporal dependency is established for a given variable, this information can be used to interpolate values for points not measured. If these variables are spatially correlated with the variable of interest, this information can be used to improve estimates. The use of auxiliary information in spatial prediction is referred to as cokriging. One of the appealing features of cokriging is that the auxiliary information does not have to be collected at the same data points as the variable of interest. This allows us to combine remote sensing and field data to provide a full coverage map with a higher resolution than would have been possible by using remote sensing or field data alone.

Prior to fitting the cokriging model, the residuals of the model describing the large-scale spatial variability are analyzed for anisotropy (spatial autocorrelation changes with direction). The residuals are also evaluated for the presence of spatial cross-correlation with the independent variables included in the large-scale model, or variables for which only data associated with field plot locations are available. If no spatial cross-correlation is detected, the residuals can be modeled using ordinary kriging, otherwise the residuals are modeled using cokriging.

The ability to spatially model field data allows integration over any specified geographical region (i.e., point- and plot-level field data, management unit, watershed, region) to obtain a point estimate and associated standard error of prediction. In essence, these types of predictive models provide information on large-scale spatial variability, while field data provides information on small-scale spatial variability as was demonstrated for exotic plants in Rocky Mountain National Park (Figure 3).

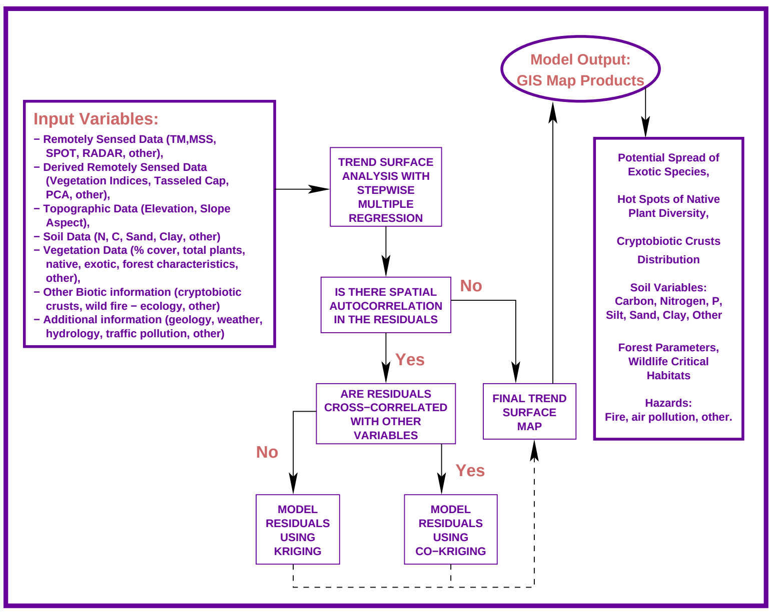


Figure 2. The development of predictive spatial models for exotic plant species, uncertainty, forest parameters, soil variables, and other biotic and abiotic factors relies on the creation of trend surface maps with stepwise multiple regression residuals using an OLS procedure.

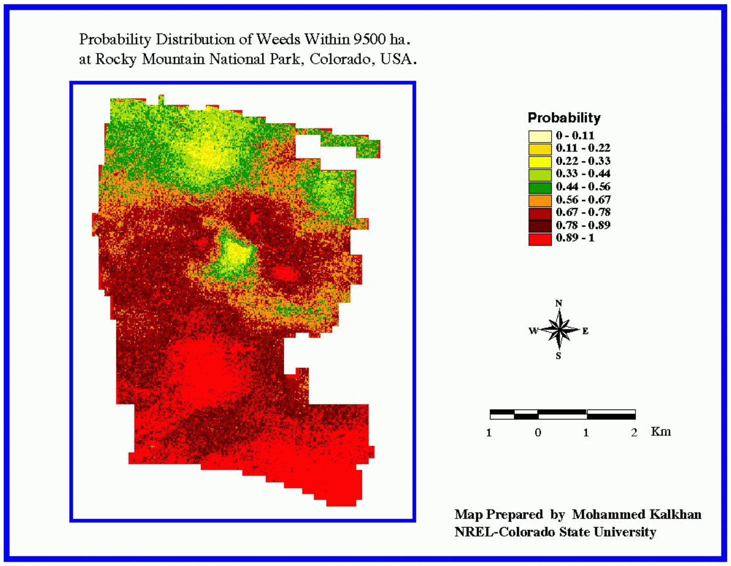


Figure 3. Example predicted probability distribution map of weeds within 9500 ha. at the Rocky Mountain National Park, Colorado, USA.

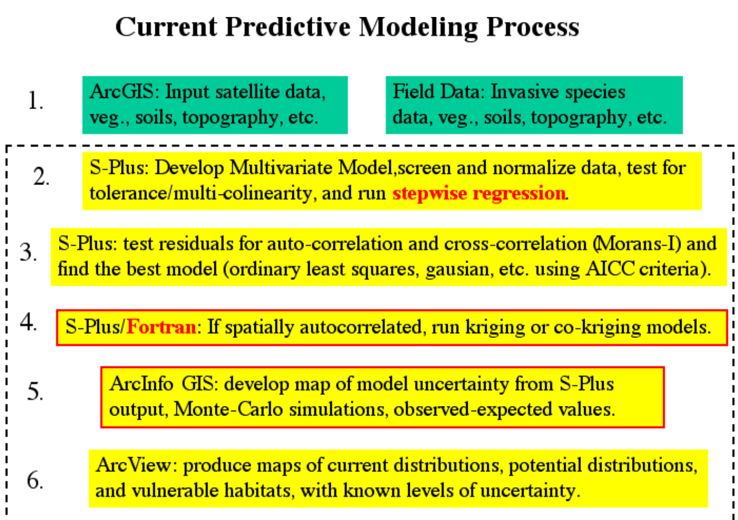


Figure 4. Current Predictive Modeling Process

The current implementation of this modeling process (Figure 4) relies on a collection of poorly integrated COTS applications. The kriging and cokriging steps (No. 4), currently performed within the statistical package S-Plus, are time-consuming and significantly limit the scaling capacity of this approach. This project will focus on improving the performance of kriging and cokriging and will encapsulate the modeling process within a comprehensive, user-friendly application framework.

3.3 Preliminary Requirements

The Invasive Species Forecasting System (ISFS) will:

1. Provide a web-accessible graphical user interface.
2. Enable the user to select from a list of available datasets or to upload new data.

   Datasets will include, at a minimum:

   (a) Nested plot survey data of plant species
   (b) Photointerpreted map species
   (c) Field samples of soil to determine total carbon, nitrogen, and soil texture and biotic variables
   (d) Digital Elevation Model (DEM) and Landsat TM data (30 x 30 m resolution).

3. Perform predictive modeling

   (a) Run a Stepwise Ordinary Least Square (OLS) regression to develop a multivariate model relating the presence of exotic species (=Y) to slope, aspect, elevation, biotic variables, soil properties, the original TM bands and indices derived from TM bands variables (=X), selecting the best model based on Akaike's Information Criteria, Corrected (AICC).
   (b) Test the hypothesis that the residuals are not spatially autocorrelated with themselves and cross-correlation with the input variables, using Moran's I test statistic.

   If there is no indication of spatial autocorrelation, use the model from the stepwise OLS regression. When there is a statistically significant indication of auto- and cross-correlation, the following steps will be performed:

   (c) Fit isotropic variogram models to the empirical variograms (and empirical cross-covariograms), considering Gaussian, spherical, and exponential models, selecting the best model based on the lowest AICC.
   (d) Krige (or cokrige when cross-correlation is present) a residual surface and include these variables in the OLS model.

4. Produce maps and reports of current distribution, potential distributions, and vulnerable habitats with known levels of uncertainty.

   (a) Interactively display maps.
   (b) Allow user to download the digital map data in a variety of forms.

4 Technical Approach

4.1 Development Environment

Figure 5 shows the general architecture we plan for our development environment. This project is a multi-agency collaboration involving scientists whose range of experience with software engineering and application development is varied. One of our goals is to help develop good software engineering practices ("lead by example") and convey NASA expertise in high-performance computing to our colleagues at USGS. It is therefore important that we maintain agility and flexibility with hardware and software options at all stages of development and be aware of the constraints and opportunities of the USGS context in which the results of this work will ultimately be deployed.

We will keep project-specific hardware to a minimum, performing most development on distributed non-dedicated resources. We plan to prototype the hardware architecture for our application at Goddard and then move it to USGS. With a flexible architecture, we can re-implement the system on a larger cluster.

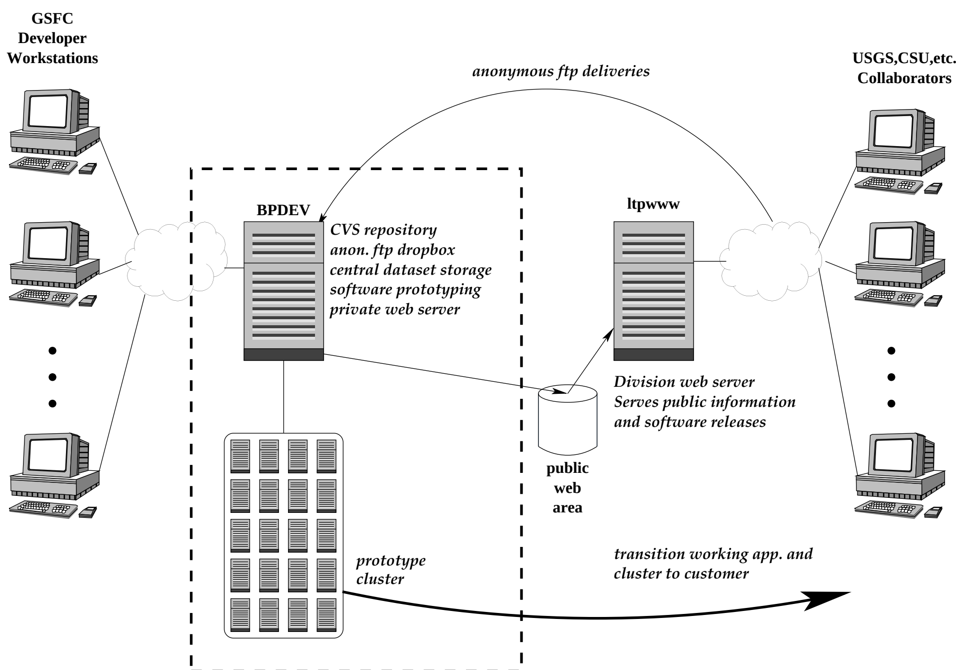


Figure 5. Hardware Development Environment

We will acquire a development computer (BPDEV) to be located at NASA Goddard Space Flight Center in Greenbelt, Maryland. We will use BPDEV to develop the *Invasive Species Forecasting System* (ISFS) as a web application that utilizes an attached commodity cluster. BPDEV will run the Linux operating system, as packaged by Red Hat, Inc. It will be professionally administered by the Laboratory for Terrestrial Physics (LTP) Computing Facility (LTPCF) system administration (SA) staff.

As development of the application warrants, we will acquire and install a prototype cluster at GSFC. Once the application is ready, we will install the application and cluster at the "customer" site at USGS.

The SA responsibilities for BPDEV will include OS upgrades/patches, security monitoring, daily backups, application software upgrades, maintenance of user accounts, etc. BPDEV will run a variety of public domain and COTS software, including CVS, the Concurrent Versions System; the Apache web server environment; Splus from Insightful; ArcInfo from ESRI, IDL, ENVI, and ION from RSI; Matlab from Mathworks; Java, FORTRAN, C, and C++ compilers; etc.

BPDEV will serve as the CVS repository, under control of the Configuration Manager (CM). Developers at CSU and USGS will deliver new software and/or data to a secure incoming ftp directory, from which the CM will transfer the material to CVS. The ncftp daemon and file permissions will be used to maintain the security of this incoming directory, and only GSFC shell users will be able to access the transferred files.

BPDEV will host a private web site and serve as the front end to the GSFC development cluster, which will enable us to build and test the cluster and the front-end web software in a highly secure environment.

We will also maintain a public web site at http://ltpwww.gsfc.nasa.gov/BP, which we will use to make our required deliveries to the CT project.

4.1.1 Security

The BP project development hosts will be covered by the Laboratory for Terrestrial Physics (LTP) security plan. All administration will be performed in accordance with NASA Procecedures and Guidelines expressed in NPG 2810.1, *Security of Information Technology*.

In particular, login access to BPDEV will only be allowed with Secure Shell (SSH), and will be restricted to specific IP addresses within the `gsfc.nasa.gov` domain.

4.2 Community Engagement

The products of this work will be useful to other scientists and to state and local land management agencies. We will schedule a series of design, review, and training meetings with agencies that collaborate with MESC on this and related projects. The clients we intended to work with include the following:

- US Department of Agriculture
- US Fish and Wildlife Service
- National Park Service
- US Forest Service
- Long-Term Ecological Research Network
- University of California at Davis
- Colorado Natural History Program
- The Nature Conservancy
- Colorado State University
- Natural Resource Ecology Laboratory
- The Biota of North America Program
- Bureau of Land Management
- Grand Staircase-Escalante National Monument
- Rocky Mountain National Park

We will also deliver products to the extended community through an established and significant community infrastructure program: USGS's National Biological Information Infrastructure (NBII).[1] NBII is a broad, collaborative program to provide increased access to data and information on the Nation's biological resources. NBII links diverse, high-quality biological databases, information products, and analytical tools maintained by NBII partners and other contributors in government agencies, academic institutions, non-government organizations, and private industry. NBII partners work on new standards, tools, and technologies that make it easier to find, integrate, and apply biological resources information. Resource managers, scientists, educators, and the general public use these programs to answer a wide range of questions related to the management, use, or conservation of this Nation's biological resources.

[1] See http://www.nbii.gov for additional information

4.3 Reuse and Reusability Strategy

Reuse is a key tenet of this project and will be approached from two directions:

1. Reuse of other work within our software application
2. Including design elements within our software to make it easy for other projects to reuse our work

4.3.1 Reuse

Previous research at CSU and USGS has resulted in numerous algorithms and software modules related to the software application we intend to develop. We also have a number of prototyping efforts already underway to incorporate and analyze the software available to us to determine its applicability and appropriateness for reuse.

In addition, we are considering a number of COTS packages and frameworks that are heavily used by the scientific community and whether or not we can build on functionality that already exists within those packages. Since a major goal of this project is to vastly improve the performance of our code using HPC techniques, the COTS packages may not be ultimately incorporated, but we may be able to take advantage of certain design elements anyway.

4.3.2 Reusability

Our software will be designed and developed so that it will be easy for our work to be reused in future software. Where possible and appropriate, we intend to develop distinct modules implementing certain functionality. By carefully considering the interfaces and fully documenting the software application program interfaces (APIs), the modules will be easier for others to reuse.

As the community does make use of a number of COTS frameworks (IDL, SPLUS, etc.), we may be able to make portions of our code accessible from those frameworks.

We will consider portability throughout the design and development, attempting to encapsulate any architecture dependencies. This will make the code easier to adapt to other systems. We intend to compile and test our software on multiple architectures to ensure this portability.

4.4 Defect Tracking

We will implement the *Bugzilla* Defect Tracking System, to track bugs and enhancement requests. *Bugzilla* allows us to enter a defect into the database, assign it to a specific individual, assign a priority and attach information about the issue (i.e. how to reproduce, analyses, plans, etc.).

Each bug is tracked through the various states of its lifecycle from newly entered to resolved and resolution verified. If a bug has a dependency (can't be fixed until some other bug is resolved), that can be noted in the database as well.

Bugzilla also provides numerous reports and makes it easy to keep track of the work planned without losing issues "through the cracks."

4.5 Software Lifecycle

The ISFS will be developed using a phased approach based on the "spiral" approach to the traditional waterfall model. This section describes our planned flow through the major development phases and our intended activities, products and documentation for each phase.

Figure 6 depicts the phases of the waterfall, the formal reviews that are planned, and the documents that will be produced. Many of the documents will be revised and extended in subsequent phases.

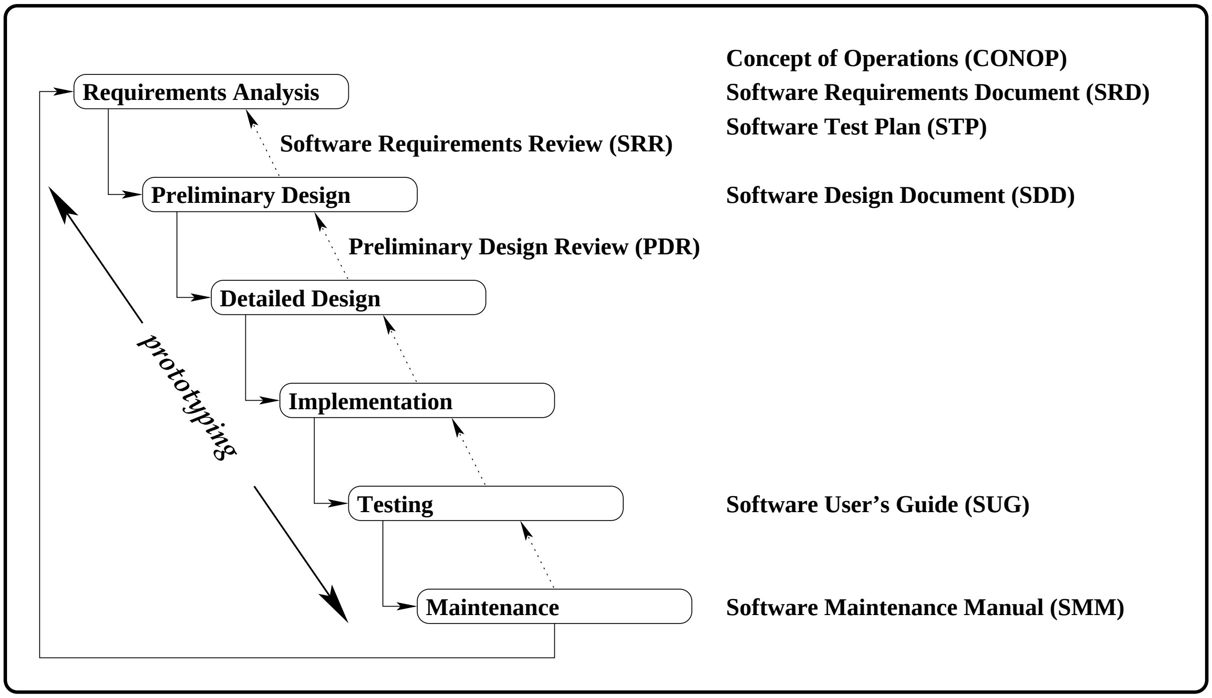


Figure 6. Software Lifecycle Waterfall

After the initial release of the software, we will revisit requirements and design, making changes as needed to add additional functionality and, in particular, increase performance.

Along with our structured approach intended to produce a high quality, polished product, we will also be performing prototyping activities to experiment with different ways of approaching the problem. We will be sharing the prototype versions of the application with our USGS collaborators and getting feedback that will be incorporated into the system design and development.

4.5.1 Requirements Analysis

We will document the operation of the system in a *Concept of Operations* (CONOP). The CONOP will included descriptions of the intended use of the software, and a break down of the software components that will be part of the final system. It will also describe the external interfaces that will be implemented by the software.

This plan includes a very high level description of the software, its objectives and high level requirements. We will decompose and analyze those requirements, deriving further detailed requirements that will be documented in a *Software Requirements Document* (SRD). The SRD will be the formal documentation of the requirements that the system produced by the project will meet.

Special efforts will be made to gather requirements from the scientific community that will potentially use the software. As discussed, a major objective of the project is to produce software that can be easily used (and components that can be re-used), requirements will be included that will reflect this objective.

We will have a formal **Software Requirements Review** (SRR) to validate the final set of requirements. The review will ensure that the accepted requirements are complete, concise, and consistent.

After the SRR, we will baseline the requirements. After that, the requirements will be placed under formal configuration management, and changes will be subject to approval by the project Configuration Control Board (CCB) (See the *Configuration Management Plan* (CMP) for more information.)

A *Software Test Plan* (STP) will be produced that describes the testing that will be performed to validate that the software meets the requirements. It will include a Verification Cross-Reference Index (VCRI) that indicates the verification methods that will be used for each requirement (inspection, analysis, demonstration, or testing) and maps test events to specific requirements.

4.5.2 Preliminary Design

Working from the SRD and CONOP, the system design will be refined into a high level view of the system and software components.

A *Software Design Document* (SDD) will be produced to capture the complete architectural description of the software. This will include a description of the functional data flow within the software and detailed documentation of the interfaces and interactions that the software will have with external entities.

The SDD will include a Requirements Traceability Matrix (RTM) that will map the requirements to the software components that will implement the functionality needed to meet the requirement. This will ensure that each requirement is fulfilled in the final software.

The SDD will be subjected to a **Preliminary Design Review** (PDR) to ensure that the software described is feasible and correct, and that it will ultimately meet the requirements.

In particular, the design will be reviewed with regard to software reuse, portability, and interoperability with other community efforts. The software will be designed for ease-of-maintenance and robust integration of experimental modules.

4.5.3 Detailed Design

The various components that are described in the SDD will be designed and developed by individuals and small teams. As more detailed design proceeds, design artifacts documenting the design of software modules will be collected and added to the SDD. In particular, we intend to capture the following (for modules where they are relevant):

- Sufficient detail to describe the architectural features, module interfaces, APIs, class hierarchies and functionality of the module.
- A description of the problem class for which the code was designed.
- An enumeration of the equations solved and their boundary conditions.
- A description of numerical methods used in the code.
- A description of parallelization techniques used.
- References to easily accessible published papers or documents relevant to the design.

4.5.4 Implementation

During implementation, source code documentation will augment the SDD to fully describe the software. Sufficient internal documentation will be used to ensure that others can understand the structure and flow of the code. This documentation will include:

- An overview of the code: key routines and data structures, and grouping of routines and data structures into functional units or reusable objects/classes.
- Comments describing the purpose of each significant data structure/variable used in the code.
- Header comments for each functional routine that conform to the NASA template guidelines.
- Internal comments within each function as required for describing key code segments or explaining subtleties of the algorithm or data representation.

4.5.5 Testing

During the design phases, the *Software Test Plan* (STP) will be elaborated and extended. It will ultimately include documentation of the detailed software testing procedures, including details of the test cases that make up each test event:

- Specify all input and output data required.
- Describe the software components and functionality that the test case verifies and the specific requirements satisfied.
- Specify the system configuration, libraries, executables, and any other test environment requirements necessary to successfully perform the test case.

As defects are noted during tested they will be entered into the Defect Tracking System described in section 4.4 on page 14.

As the user's operational interface is tested, a *Software User's Guide* (SUG) will also be developed.

4.5.6 Maintenance

We intend to maintain the project web site indefinitely, including all the project documentation and at least the latest software release baseline. All software versions can be retrieved from the software configuration management tool (CVS) as needed.

Major software documents and releases of the software delivered to the ESTO/CT Project will also be hosted on their project web site permanently.

A separate *Software Maintenance Manual* (SMM) will complement the *Software Design Document* (SDD). It will provide sufficient detail to allow maintenance of the source code, implementation of new features, and enhancements to the functionality of the software system.

4.6 Build strategy

We plan to develop the software development through several build iterations. Each build will either add functionality to the software, or increase the performance of the software.

The major build milestones are:

MILESTONE E	Code Baseline Completed
MILESTONE F	First Code Improvement Completed
MILESTONE G	Second Code Improvement Completed

As described above, we will be developing a comprehensive test suite to verify functionality. We will perform regression testing at each major milestone to ensure that no previously demonstrated functionality is lost. We will capture detailed performance statistics to analyze the performance improvements. We will also thoroughly document the hardware architecture used for each test.

As the code is baselined and preliminary design is performed, we will refine our build strategy and determine the functionality we expect to include with each major build.

4.7 Tools and Technologies

Throughout this year we are exploring several techniques for ingesting data and verifying the validity of the inputs to the system. We will define and implement the human interface structure and elements that will comprise the GUI. Our baseline code will be re-written to take advantage of multiple processors.

Our software tools, wherever possible will be free or open source when such options are available. Such tools are generally more cost effective, secure, and standards compliant than their commercial counterparts.

We fully expect to use the power of a relational database management system (RDBMS) to capture metadata and to ensure that data points fall within accepted ranges. Our database will be at the core of every functional element of our system and will be the foundation upon which each module will be built. Our prime candidate is the UC Berkeley derived PostgreSQL database (available from http://www.postgresql.org). Our reasons for choosing PostgreSQL include but are not limited to:

- Stability
- The rich set of built in data types (including geometric and network)
- The capability for creating custom data types
- Tight language integration with C, C++, Perl, Python, and procedural languages
- SQL92 compliance
- Highly configurable access control
- Ease of administration

Our web server is an Apache HTTP Server (available from http://www.apache.org) The Apache httpd server is a powerful, flexible, HTTP/1.1 compliant web server that implements the latest protocols. It is highly configurable and extensible with third-party modules. It s also easily integrated with many popular server side integration options including mod_perl. It comes with useful documentation, full source code, and an unrestrictive license.

Our interface building components are still being evaluated, but we will probably use a mix of:

- Java (available from http://www.sun.com),
- Python (available from http://www.python.org),
- JavaScript (described in http://wp.netscape.com/eng/mozilla/3.0/handbook/javascript)
- CGI (described in http://hoohoo.ncsa.uiuc.edu).

All of these languages and scripting tools are ideal for creating and customizing web content and applications (GUI) delivered through the web server to browser enabled clients. All these tools used in concert will give the interface the power, flexibility, and customization options that our users will expect. The user preferences and parameters will of course reside in and be served by our PostgreSQL RDBMS.

Our modeling components will include:

- IDL, the Interactive Data Language
- ENVI, the Environment for Visualizing Images
- ION, IDL On the Net

(all available from http://www.rsinc.com) as they are ideally suited to the analysis and display of geostatistical information.

We are continuing to explore the following technologies:

- S-Plus and FORTRAN to develop our statistical analysis and visualization methods (available from http://www.insightful.com/products/product.asp?PID=10).
- We have evaluated the ESRI tools and have found them to be both too expensive and feature laden to be included as part of the second year effort. We will try to use GRASS in lieu of the ArcInfo suite to perform the mapping functions and coordinate system transformations.
- Our hardware will be commodity X86 based computers with fast Ethernet and large hard drives that will work in the Beowulf (http://www.beowulf.org/) cluster architecture.
- GRASS GIS (Geographic Resources Analysis Support System) (available from http://www3.baylor.edu/grass/index2.html) as a tool to answer our map creation requirement. GRASS GIS is an open source, free software Geographical Information System (GIS) with raster, topological vector, image processing, and graphics production functionality that operates on various platforms through a graphical user interface and shell in X-Windows. It is released under GNU General Public License (GPL).
- We will be working with Jakarta Tomcat. Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet (http://java.sun.com/products/servlet/index.html) and JavaServer Pages (http://java.sun.com/products/jsp/) technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process (http://jcp.org/).

5 Management Approach

5.1 Schedule

Figure 7 is the overall high level schedule for the project. More details of specific tasks are in appendix C “Task Descriptions” and appendix D “Schedule Detail”.

ID	Task Name	Start
1	**BP Project**	Tue 1/29/02
2	Project Start	Tue 1/29/02
3	Software Engineering Plan completed	Mon 4/15/02
4	Code baseline completed	Mon 7/15/02
5	First Annual Report delivered	Thu 8/15/02
6	First code improvement completed	Tue 7/15/03
7	Linux clusters installed	Tue 7/15/03
8	Second Annual Report delivered	Fri 8/15/03
9	Second code improvement completed	Thu 7/15/04
10	Third Annual Report delivered.	Mon 8/16/04
11	Customer delivery accomplished.	Wed 12/15/04
12	Final Report delivered	Tue 3/15/05

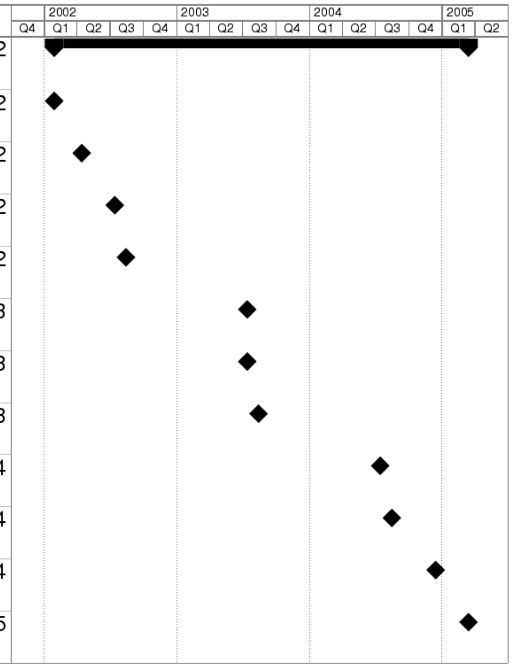


Figure 7. Schedule

5.2 Evaluation

This project involves a relatively small, tightly knit development team. Ongoing progress will be monitored by the Project Manager (PM) through periodic status meetings and informal reports. In addition to this informal evaluation, the schedule includes numerous formal reviews and milestones for major deliveries. Special management attention will be used to manage to the schedule to ensure that the milestones will be met.

5.3 Risk Management

We intend to implement a formal risk management strategy for identifying, assessing, analyzing, monitoring and mitigating risks.

The risk management strategy is described in the *Risk Management Plan* (RMP).

5.4 Validation Plan

The testing regimen and other verification activities will ensure that the software as developed meets the requirements defined for it. We also plan to implement various validation activities to ensure that the software meets the needs of the customer and performs as expected.

We will involve the customer (user) community early in the software development process, through our formal requirements and design reviews. This will validate our baslined requirements and design, keeping us on course to produce software that will be valuable and useful to the customer. The customer will also be provided with early prototype versions of the software and hands on training with the software as applied to real world problems.

We will be holding an annual workshop to present the software and specific applications using it to an even wider customer audience. This will give us direct feedback into the needs of the customer and ensure that we incorporate them into the design of the software.

6 Product Assurance

6.1 Configuration Management

We will use formal configuration management techniques to control and manage the changes that are inevitable in any project.

Details can be found in the *Configuration Management Plan* (CMP).

6.2 Quality Assurance

We will use a variety of Quality Assurance techniques to maintain consistently high quality in the software products we produce.

Details can be found in the *Quality Assurance Plan* (QAP).

7 Plan Update History

7.1 Version 1.2, September 26, 2002

Added several more team members to appendix A: Robert Baker, David Herring, David Kendig and David Obler.

Added appendix C “Task Descriptions” and appendix D “Schedule Detail”.

Added section 4.7 “Tools and Technologies”.

A Contact Information

A.1 Team Members

Robert Baker
SSAI
10210 Greenbelt Road
Suite 500
Lanham, MD 20706
Phone: +1 301-867-2073
Fax: +1 301-867-2191
Email: `Robert_Baker@sesda.com`

John E. Dorband, PhD
NASA Goddard Space Flight Center
Building 28, Room S206
Code 935
Greenbelt, MD 20771
Phone: +1 301-286-9419
Fax: +1 301-286-1634
Email: `John.E.Dorband.1@gsfc.nasa.gov`

Michael T. Frame
Biological Resources Division
US Geological Survey
Reston, VA 20192
Phone: +1 703-648-4164
Fax: +1 703-648-4224
Email: `mike_frame@usgs.gov`

David Herring
NASA Goddard Space Flight Center
Building 33, Room A325
Code 913.0
Greenbelt, MD 20771
Phone: +1 301-614-6219
Fax: +1 301-614-6307
Email: `David.D.Herring@gsfc.nasa.gov`

Mohammed A. Kalkhan, PhD
Colorado State University
Natural Resource Ecology Laboratory
NREL-A244
Fort Collins, CO 80523-1499
Phone: +1 970-491-5262
Fax: +1 970-491-1965
Email: `mohammed@nrel.colostate.edu`

David Kendig
NASA Goddard Space Flight Center
Global Change Master Directory
Code 902
Greenbelt, MD 20771
Phone: +1 301-867-2084
Fax: +1 301-614-6695
Email: `David.J.Kendig@gsfc.nasa.gov`

Jacqueline J. Le Moigne, PhD
NASA Goddard Space Flight Center
Building 28, Room W186
Code 935
Greenbelt, MD 20771
Phone: +1 301-286-8723
Fax: +1 301-286-1777
Email: `lemoigne@backserv.gsfc.nasa.gov`

Jeffrey T. Morisette, PhD
NASA Goddard Space Flight Center
Building 33, Room G325
Code 923
Greenbelt, MD 20771
Phone: +1 301-614-6676
Fax: +1 301-614-6695
Email: `Jeffrey.T.Morisette.1@gsfc.nasa.gov`

David Obler
SSAI
10210 Greenbelt Road
Suite 500
Lanham, MD 20706
Phone: +1 301-867-2151
Fax: +1 301-867-2191
Email: `David.Obler@gsfc.nasa.gov`

Jeffrey A. Pedelty, PhD
NASA Goddard Space Flight Center
Building 33, Room G325
Code 923
Greenbelt, MD 20771
Phone: +1 301-614-6609
Fax: +1 301-614-6695
Email: `Jeffrey.A.Pedelty.1@gsfc.nasa.gov`

Robin M. Reich, PhD
Department of Forst Sciences
Colorado State University
Ft. Collins, CO 80523
Phone: +1 970-491-6980
Fax: +1 970-491-6754
Email: `robin@cnr.colostate.edu`

John L. Schnase, PhD
NASA Goddard Space Flight Center
Building 28, Room W230D
Code 930
Greenbelt, MD 20771
Phone: +1 301-286-4351
Fax: +1 301-286-1777
Email: `schnase@gsfc.nasa.gov`

James A. Smith, PhD
NASA Goddard Space Flight Center
Building 33, Room G125D
Code 920
Greenbelt, MD 20771
Phone: +1 301-614-6020
Fax: +1 301-614-6015
Email: `James.A.Smith.1@gsfc.nasa.gov`

Thomas J. Stohlgren, Phd
Midcontinent Ecological Science Center
US Geological Survey
Ft. Collins, CO 80523
Phone: +1 970-491-1980
Fax: +1 970-491-1965
Email: `toms@nrel.colostate.edu`

Curt A. Tilmes
NASA Goddard Space Flight Center
Building 32, Room S36C
Code 922
Greenbelt, MD 20771
Phone: +1 301-614-5534
Fax: +1 301-614-5269
Email: `Curt.Tilmes@gsfc.nasa.gov`

A.2 Document / System Access

ESTO/CT deliverables for this project are available at http://ltpwww.gsfc.nasa.gov/BP/deliverables.html. The baseline system along with complete documentation are available on the project's BPDEV computer (`frio.gsfc.nasa.gov`). Users may log on to the system to run the baseline program (please contact John Schnase at 6-4351 for userid and password). In addition, a tarfile is available from both the website and the ISFS home directory that can be used to build the baseline environment on a different machine.

B Glossary

AICC Akaike's Information Criteria, Corrected
API Application Program Interface
BP Biotic Prediction project
BPDEV Biotic Prediction Development Host
CCB Configuration Control Board
CI Co-Investigator
CT Computational Technologies project
CM Configuration Manager / Configuration Management
CMP Configuration Management Plan
CONOP Concept of Operations
COTS Commercial Off The Shelf
CSU Colorado State University
CVS Concurrent Versions System
ESTO Earth Science Technology Office
FORTRAN FORmula TRANslator computer language
GSFC Goddard Space Flight Center
IDL Interactive Data Language
ION IDL On the Net
IP Internet Protocol
ISFS Invasive Species Forecasting System
LTP Laboratory for Terrestrial Physics
LTPCF Laboratory for Terrestrial Physics Computing Facility
NBII National Biological Information Infrastructure
NPG NASA Procedures and Guidelines
NREL Natural Resources Ecology Laboratory
OLS Ordinary Least Square regression technique
OS Operating System
PDR Preliminary Design Review
PI Principal Investigator
RMP Risk Management Plan
RSI Research Systems Incorporated
RTM Requirements Traceability Matrix
SA System Administration
SDD Software Design Document
SEP Software Engineering / Development Plan
SMM Software Maintenance Manual
SRD Software Requirements Document
SRR Software Requirements Review
SSH Secure Shell
SUG Software User's Guide
TBD To Be Determined or To Be Developed
USGS United States Geological Survey
QAP Quality Assurance Plan

C Task Descriptions

Table 3. Task Descriptions

ID #	Task Name
1	**INGEST- Validation**
2	The system shall verify integrity, but not necessarily validate the quality of the data before ingest.
3	**INGEST - Field Data**
4	The system shall provide standard templates for ingesting field data in a tabular form.
5	The templates shall include all required fields to be captured.
6	The templates shall be in an accessible format (such as spreadsheet, database, or simple ASCII
7	**INGEST - Remote Sensing Data**
8	The system shall support ingest from external satellite data archives (such as the Goddard DAAC).
9	The system shall support ingest of user-supplied satellite data or airborne imagery from digital files.
10	The system shall support ingest of user-supplied layers.
11	The system shall support ingest of user-supplied data files for ancillary layers.
12	**INGEST - Data Acquisition**
13	The system shall provide accounting and logging by requiring users' name and password.
14	Test the format for the tabular data to include tab and white space delimited tables, along with comma delimited
15	Test and evaluate only allowing one tabular data set for each model run
16	Convert all image data sets to GeoTiff as a common data format
17	**PRE-PROCESSING - Merge Data**
18	Test the extraction of values from the raster data sets for each location given in the tabular data
19	Begin a test to append the extracted raster values to the tabular data sets with proper column headings indicating which raster file and layer was used to create the column
20	Begin evaluating the conversion of raster data sets to GeoTiff format from the Ingest phase
21	Begin testing re-sampling of the raster data so that all raster data are the same spatial resolution, keeping a copy of the original and re-sampled data
22	**MODELING**
23	The system shall allow the ability to specify response and explanatory variables from the available databases.
24	The system shall have the ability to ”fit” models through least squares (or other) optimization routines.
25	The system shall provide screening techniques to quantitatively assess which explanatory variables are related to the response variable (such as stepwise regression).
26	The system shall calculate geospatial statistics (such as variograms).
27	The system shall be able to output model results and relevant model diagnostics.
28	**POST-PROCESSING Reprojecting Data**
Table 3 continued	

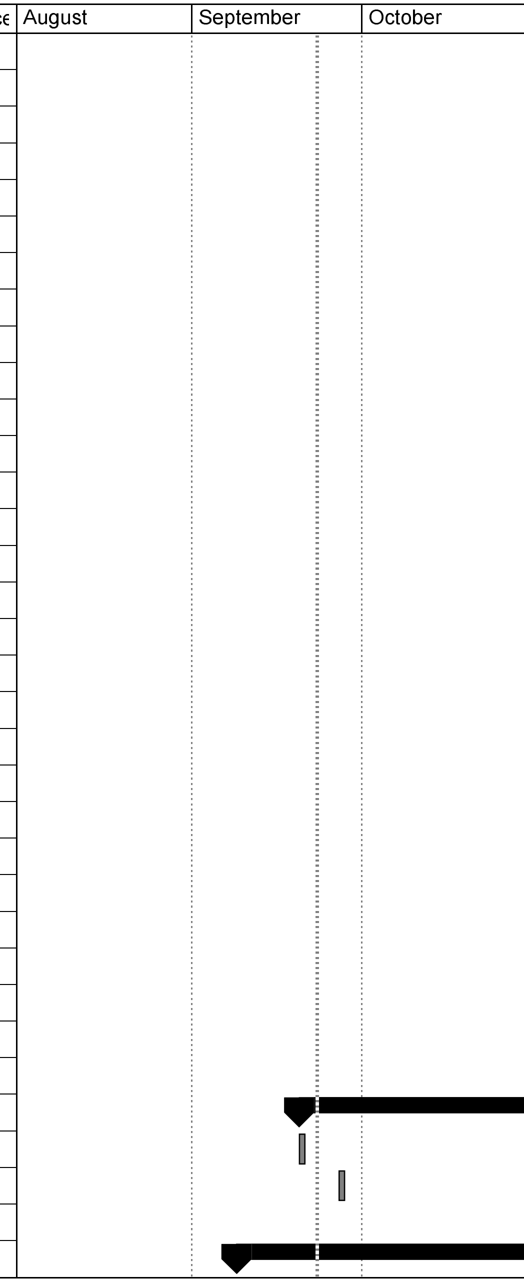
continued Table 3	
29	The system shall produce a data file suitable for reprojection by external COTS utilities.
30	**USER INTERFACE Graphical User Interface**
31	Install and configure Bugzilla tracking system for tracking development bugs
32	Install and configure CVS for software version control throughout development
33	The system shall include a Graphical User Interface (”GUI”) to support user interaction with the system.
34	**USER INTERFACE Baseline Presentation of Concept (POC)**
35	A working prototype of the system will run the established baseline from a browser window
36	The prototype will launch a gas gauge when the baseline is executed to show the process is running
37	When the baseline process is completed, the prototype will be able to launch the resultant image in a new window
38	A review of the ISFS prototype by the potential user group sill provide valuable feedback on system features, layout, design, and expectations
39	**MILESTONE F Delivery Requirements**
40	Configure and test frio as part of the medusa cluster to run a process on 32 nodes
41	Run prototype kriging in parallel on medusa with error-checking for final output of kriged surface
42	Create version 1.0 of parallel kriging code for final output image
43	Evaulate and tune performance of vers. 1.0 of kriging code for asynchronous transfer
44	Design, construct and deploy the Linux Cluster at Colorado Sate University
45	Deliver products 25X than the baseline implementation
46	Create scaling curves for parallel kriging code to increase the number of processors
47	Update the Design Document with latest changes noted in draft form
48	Distribute updated Design Document Draft to team members for review
49	Review Updated Design Document Draft and note corrections and changes
50	Finalize and deliver updated Design Document
51	Update the Requirements Document with latest changes noted in draft form
52	Distribute updated Requirements Document Draft to team members for review
53	Review Updated Requirements Document and note corrections and changes
54	Finalize and deliver Updated Requirements Document
55	Write draft of initial Test plan/procedures document
56	Distribute draft of test plan to team members for review
57	Review draft of Test plan/procedures document
58	Deliver initial Test plan/procedures document
59	Make all documented source code available via the projects web site
60	**MILESTONE F - Project Milestones**
61	Baseline Presentation of Concept (POC) is completed and can be presented to demonstrate usability of the system as a browser based solution
62	Version 2.0 kriging code fine tuned and completed and ready for deployment as a system solution
Table 3 continued	

continued Table 3	
63	Browser to back-end connectivity achieved allowing the user to login and successfully upload datasets to the system, run those datasets, and view the results as image(s)
64	Model runs successfully using browser and medusa cluster, paving the way for implementation in the CSU (Colorado State Universtity) cluster
65	Linux Cluster @ CSU running and processing as specified
66	ISFS performing as specified, expectations achieved at 25x the baseline implementation
67	Milestone F - All Documentation Completed and ready for distribution via project web site
68	Milestone F - All Requirements Delivered

D Schedule Detail

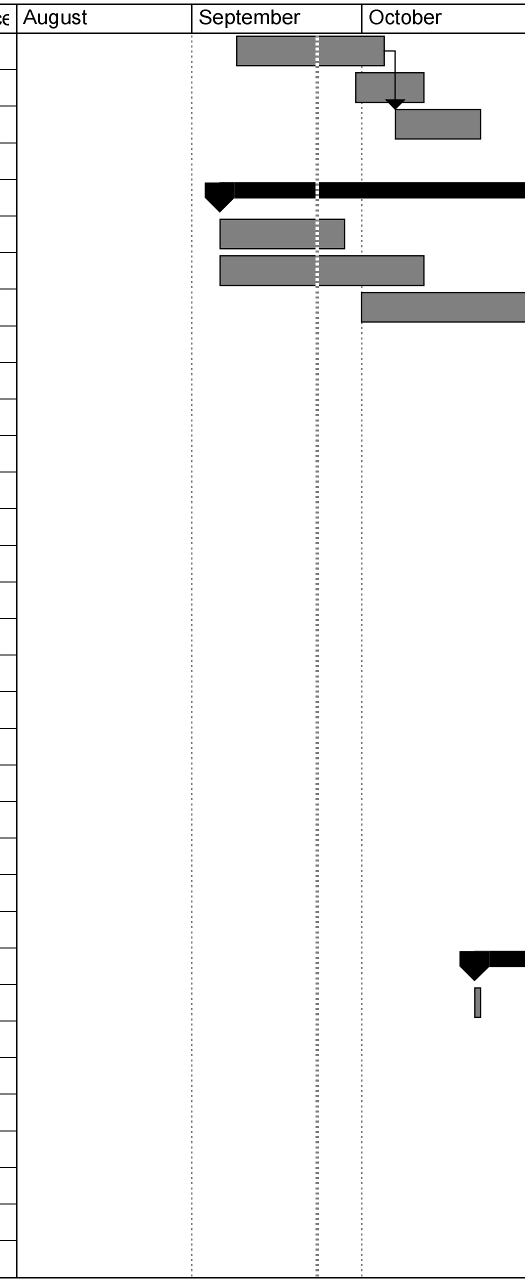
See schedule on next 4 pages.

ID	Task Name	Duration	Start	Finish	Predece
1	**INGEST - Validation**	**25 days?**	**Mon 1/6/03**	**Fri 2/7/03**	
2	Verify Integrity	25 days?	Mon 1/6/03	Fri 2/7/03	
3	**INGEST - Field Data**	**25 days?**	**Mon 1/6/03**	**Fri 2/7/03**	
4	Standard Data Templates Provided	25 days?	Mon 1/6/03	Fri 2/7/03	
5	Templates incl. All req'd. fileds	25 days?	Mon 1/6/03	Fri 2/7/03	
6	Assure templates are accessible	5 days?	Mon 2/3/03	Fri 2/7/03	
7	**INGEST - Remote Sensing Data**	**22 days?**	**Thu 1/2/03**	**Fri 1/31/03**	
8	Support user-supplied satellite data	22 days?	Thu 1/2/03	Fri 1/31/03	
9	Support airborne imagery from digital files	22 days?	Thu 1/2/03	Fri 1/31/03	
10	Support user-supplied layers	22 days?	Thu 1/2/03	Fri 1/31/03	
11	Support ingest of user-supplied data for ancillary layers	22 days?	Thu 1/2/03	Fri 1/31/03	
12	**INGEST - Data Acquisition**	**33 days?**	**Mon 11/4/02**	**Wed 12/18/02**	
13	Accounting and logging will require users' name and password	25 days?	Mon 11/4/02	Fri 12/6/02	
14	Test the format for the tabular data	4 days?	Mon 12/9/02	Thu 12/12/02	
15	Test only allowing one tabular data set for each model run	5 days?	Mon 12/9/02	Fri 12/13/02	
16	Convert image data sets to GeoTiff	3 days?	Mon 12/16/02	Wed 12/18/02	
17	**PRE-PROCESSING - Merge Data**	**30 days?**	**Mon 12/9/02**	**Fri 1/17/03**	
18	Test extracting values from data sets	5 days?	Mon 12/9/02	Fri 12/13/02	
19	Test appending extracted raster values	6 days?	Fri 12/13/02	Fri 12/20/02	
20	Evaluate data sets conversion to GeoTiff	5 days?	Mon 1/6/03	Fri 1/10/03	
21	Test resampling of the raster data	6 days?	Fri 1/10/03	Fri 1/17/03	
22	**MODELING**	**70 days?**	**Mon 1/13/03**	**Fri 4/18/03**	
23	Allow ability to specify variables from available databases	70 days?	Mon 1/13/03	Fri 4/18/03	
24	"fit" models through least squares (or other) optimization routines	70 days?	Mon 1/13/03	Fri 4/18/03	
25	Provide screening techniques to assess related varaibles	70 days?	Mon 1/13/03	Fri 4/18/03	
26	Calculate geospatial statistics	70 days?	Mon 1/13/03	Fri 4/18/03	
27	Output model results and relevant model diagnostics	70 days?	Mon 1/13/03	Fri 4/18/03	
28	**POST-PROC. - Reprojecting Data**	**77 days?**	**Thu 1/2/03**	**Fri 4/18/03**	
29	Produce a data file suitable for reprojection by COTS utilities	77 days?	Thu 1/2/03	Fri 4/18/03	
30	**USER INTERFACE - Graphical User Interface**	**121 days?**	**Fri 9/20/02**	**Fri 3/7/03**	
31	Install and configure Bugzilla	1 day?	Fri 9/20/02	Fri 9/20/02	
32	Install and configure CVS	1 day?	Fri 9/27/02	Fri 9/27/02	
33	Include a GUI to support user interaction	28 days?	Wed 1/29/03	Fri 3/7/03	
34	**USER INTERFACE - Baseline POC**	**45 days?**	**Mon 9/9/02**	**Fri 11/8/02**	



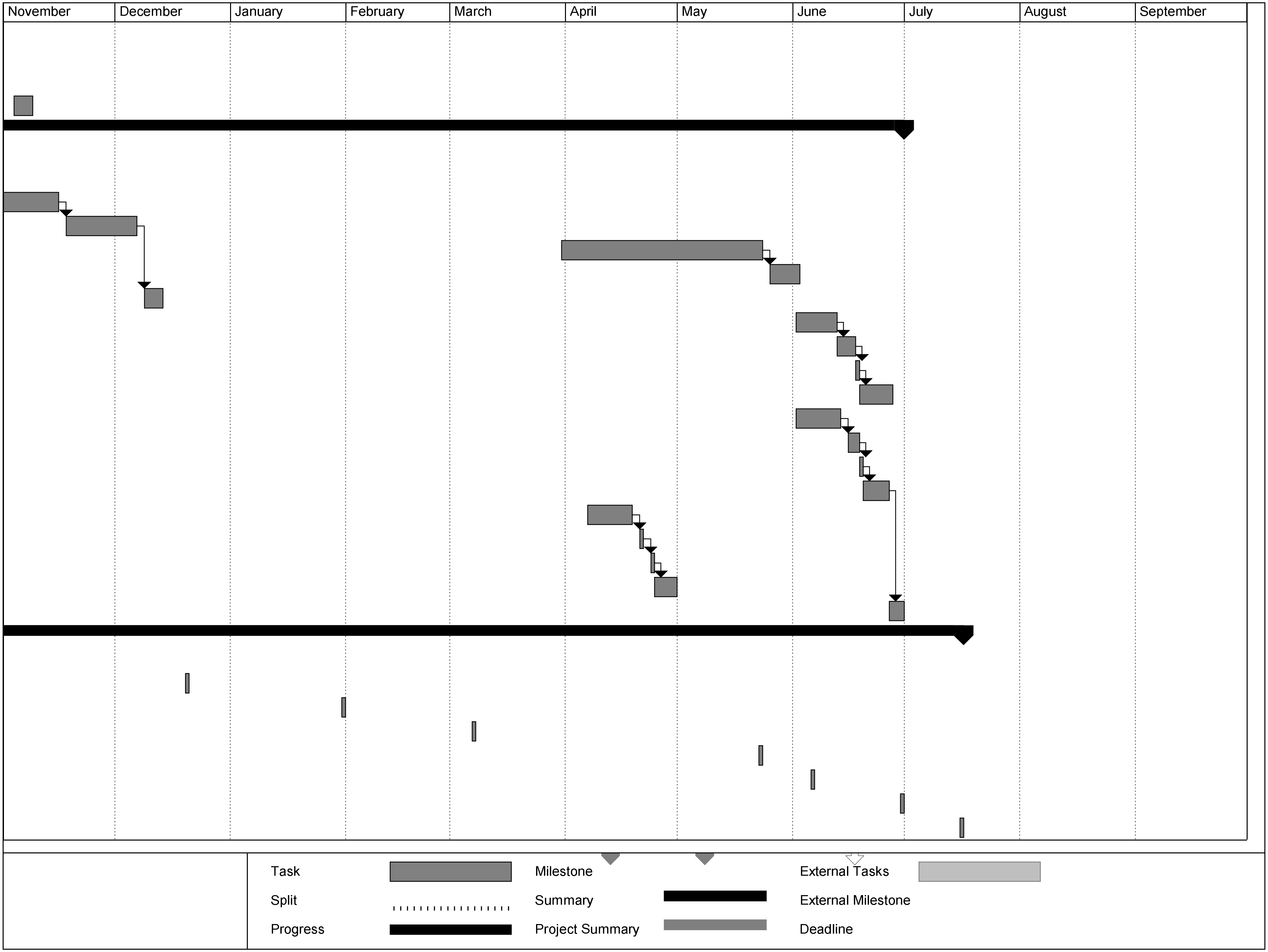
Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline

ID	Task Name	Duration	Start	Finish	Predece
35	Run baseline from browser for prototype	20 days?	Mon 9/9/02	Fri 10/4/02	
36	Display "gas guage" to show that process is running	10 days?	Mon 9/30/02	Fri 10/11/02	
37	Launch the image result in a new window	11 days?	Mon 10/7/02	Mon 10/21/02	35
38	End-user review	5 days?	Mon 11/4/02	Fri 11/8/02	
39	**MILESTONE F Delivery Requirements**	**212 days?**	**Fri 9/6/02**	**Mon 6/30/03**	
40	Configure and test frio as part of the medusa cluster	16 days?	Fri 9/6/02	Fri 9/27/02	
41	Run prototype kriging on medusa	26 days?	Fri 9/6/02	Fri 10/11/02	
42	Create version 1.0 of parallel kriging code	34 days?	Tue 10/1/02	Fri 11/15/02	
43	Evaulate and tune performance of vers. 1.0 of kriging code	15 days?	Mon 11/18/02	Fri 12/6/02	42
44	Implement Linux Cluster at Colorado Sate University	40 days?	Mon 3/31/03	Fri 5/23/03	
45	Deliver products 25X than the baseline implementation	6 days?	Mon 5/26/03	Mon 6/2/03	44
46	Create scaling curves for parallel kriging code	5 days?	Mon 12/9/02	Fri 12/13/02	43
47	Update Design Document	9 days?	Mon 6/2/03	Thu 6/12/03	
48	Distribute updated Design Document for Review	3 days?	Fri 6/13/03	Tue 6/17/03	47
49	Review Updated Design Document	1 day?	Wed 6/18/03	Wed 6/18/03	48
50	Finalize and deliver updated Design Document	7 days?	Thu 6/19/03	Fri 6/27/03	49
51	Update Requirements Document	10 days?	Mon 6/2/03	Fri 6/13/03	
52	Distribute updated Requirements Document for review	3 days?	Mon 6/16/03	Wed 6/18/03	51
53	Review Updated Requirements Document	1 day?	Thu 6/19/03	Thu 6/19/03	52
54	Finalize and deliver Updated Requirements Document	5 days?	Fri 6/20/03	Thu 6/26/03	53
55	Write draft of initial Test plan/procedures document	10 days?	Mon 4/7/03	Fri 4/18/03	
56	Distribute draft of test plan	1 day?	Mon 4/21/03	Mon 4/21/03	55
57	Review draft of Test plan/procedures document	1 day?	Thu 4/24/03	Thu 4/24/03	56
58	Deliver initial Test plan/procedures document	4 days?	Fri 4/25/03	Wed 4/30/03	57
59	Documented source code available via the Web	2 days?	Fri 6/27/03	Mon 6/30/03	54
60	**MILESTONE F - Project Milestones**	**193 days?**	**Mon 10/21/02**	**Wed 7/16/03**	
61	Baseline Presentation of Concept (POC) finished	1 day?	Mon 10/21/02	Mon 10/21/02	
62	Version 2.0 kriging code fine tuned and completed	1 day?	Fri 12/20/02	Fri 12/20/02	
63	Browser to back-end connectivity achieved	1 day?	Fri 1/31/03	Fri 1/31/03	
64	Model runs successfully using browser and medusa cluster	1 day?	Fri 3/7/03	Fri 3/7/03	
65	Linux Cluster @ CSU running and processing	1 day?	Fri 5/23/03	Fri 5/23/03	
66	ISFS performing as specified, expectations achieved	1 day?	Fri 6/6/03	Fri 6/6/03	
67	Milestone F - All Documentation Complete	1 day?	Mon 6/30/03	Mon 6/30/03	
68	Milestone F - All Requirements Delivered	1 day?	Wed 7/16/03	Wed 7/16/03	



Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline


November
December
January
February
March
April
May
June
July
August
September
Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline


November
December
January
February
March
April
May
June
July
August
September
Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline